Thai **J**ournal of **M**athematics : (2020) 75–82
Special Issue: The 14^{th} IMT-GT ICMSA 2018

http://thaijmath.in.cmu.ac.th
Online ISSN 1686-0209

Using PBIL to Minimize Makespan for Parallel Machines Scheduling Problem with Job Sequence Dependent Setup Time

Pensiri Sompong[†,1]

[†]Department of General Science, Faculty of Science and Engineering,
Kasetsart University Chalermphrakiat Sakon Nakhon Province Campus,
Sakon Nakhon 47000, Thailand
e-mail : pensiri.so@ku.th

Abstract : Parallel machines scheduling problem with job sequence dependent setup time is studied. The objective is to determine job schedule in which makespan is minimum. The problem is divided into two parts, assigning n independent jobs to m parallel machines and sequencing jobs on each machine. Population-based incremental learning (PBIL) algorithm is used to assign jobs to machines and SPT regarding sequence dependent setup time is then applied to determine sequence of job on each machine. The performance and efficiency of proposed algorithm are shown by the experiments. The solutions obtained from applying PBIL combined with SPT are compare to solutions obtained from using SPT for parallel machines. The average relative percentage deviation is 13.55% indicating good performance. From the study results, it is shown that the proposed algorithm is useful and efficient for parallel machines scheduling problem with job sequence dependent setup time.

Keywords : parallel machines; scheduling; population-based incremental learning; sequence dependent setup time.
2010 Mathematics Subject Classification : 90C59; 68M20.

1 Introduction

The parallel machines system has been widely used in manufacturing situation. In parallel machines scheduling problem, n independent jobs are assigned to be processed by only one of m parallel machines. After processing a job on machine, the time to prepare for processing the next job is performed, called setup time. In practice, setup time is sequence dependent. By this situation, parallel machines scheduling problem has been extensively studied to develop a method for determining job schedule satisfying interested performance criteria. Arnaout et al. [1] applied ant colony optimization algorithm for unrelated parallel machine scheduling problem to minimize makespan. The setup time depend not only on the job but also on the machine. Vallada and Ruiz [2] studied an unrelated parallel machines scheduling problem with machine and job sequence dependent setup times. Genetic algorithm in which local search is included in the algorithm to enhance crossover operator is used to solve the problem. The objective of

[1]Corresponding author.

the study is to minimize makespan. Li et al. [3] solved an identical parallel machines scheduling problem to minimize makespan using simulated annealing algorithm. The processing time is considered as linear decreasing functions of the consumed resource. Minimization of makespan for unrelated parallel machines with limited human resources was addressed by Cappadonna [4]. Mixed integer linear programming was used for optimally solving the problem and genetic algorithm was presented for larger size problem. Yeh et al. [5] proposed simulated annealing algorithm and genetic algorithm for parallel machines scheduling problem with learning effects to minimize makespan. Processing time was considered as fuzzy numbers to satisfy reality. An immune-inspired algorithm was proposed by Diana et al. [6] for the problem of minimising makespan on unrelated parallel machines with sequence and machine dependent setup times.

Population-based incremental learning (PBIL) algorithm is a method that combines the mechanism of genetic algorithm and competitive learning. PBIL algorithm uses probability vector to define a population and its objective is to create high values of probability in probability vector representing a population of high evaluation solution (Baluja [7]). PBIL algorithm has been used to solve optimization problem and some applications of PBIL can be found in previous work. Pang et al. [8] proposed adaptive PBIL algorithm based on analyzing the traditional PBIL algorithm. Computational tests were performed with flow shop and job shop scheduling problem. Shanshan and Dongwei [9] applied advanced PBIL algorithm to vehicle routing optimization problem. The objective is to minimize the cost and meet the time restriction. Chen et al. [10] used PBIL for cloud computing resource scheduling problem. Meng et al. [11] presented PBIL approach for serial colored traveling salesman problem to select the cities with small penalty values.

Due to many applications of parallel machines, the scheduling problem of parallel machines with job sequence dependent setup time is studied in this paper. The objective is to minimize makespan and PBIL algorithm combined with shortest processing time (SPT) is proposed to solve the problem. In section 2, the characteristics of problem, SPT for parallel machines and procedure of PBIL algorithm are addressed. PBIL applied to parallel machines scheduling problem is proposed in section 3. The performance and efficiency of proposed algorithm are discussed in section 4 and the results of the study are concluded in section 5.

2 Preliminaries

In this section, the notations and parameters used to generate the problem for the experiments are defined. In addition, the characteristics of problem and procedures of SPT and PBIL associated with parallel machines scheduling problem are addressed.

N : number of jobs.

M : number of machines such that $M = \left[\dfrac{N}{\gamma} + 0.5\right]$, γ is ratio of the number of jobs to the number of machines and $[x]$ is the greatest integer less than x.

p_j : processing time of job j, where $j \in \{1, 2, ..., N\}$.

s_{ij} : setup time between job i and job j, where $i, j \in \{1, 2, ..., N\}$ and $i \neq j$

c_{jm} : completion time of job j on machines m, where $m \in \{1, 2, ..., M\}$.

C_{max} : maximum completion time or makespan.

$\mathbf{P}$: probability vector.

$prob_k$: probability of k-th position in probability vector $\mathbf{P}$.

L : length of solution

t : number of iterations

LR : learning rate.

The characteristics of parallel machines scheduling problem discussed in this study are as follows :

(1) A machine can perform only one job at a time.

(2) Job j can be processed on any machine and it is completed by only one machine without interruption.

(3) Job j can be started immediately after job i is completed such that setup time s_{ij} is included in processing time of job j.

(4) Setup time for the first job on each machine is zero.

(5) All jobs are available to process at time zero.

2.1 Shortest Processing Time

In general, shortest processing time (SPT) has been applied for machines scheduling problem to minimize job tardiness. The sequence of jobs depends on processing time, the job with smaller processing time is firstly chosen to schedule. For parallel machines scheduling problem, SPT is utilized to minimize maximum completion time or makespan. The job with smaller processing time is selected to process on a machine having the least sum of completion time. In the case that sequence dependent setup time is considered, SPT is modified. The job with smaller processing time is assigned to be the first job on each machine and setup time of consecutive job is included in its processing time. The job causing minimum total completion time is scheduled after the predecessor job is completed.

Example 2.1. Let p and s be matrices of processing time and setup time, repectively, such that

$$p = [61\ 47\ 35\ 83\ 58\ 55], \quad s = \begin{bmatrix} - & 7 & 2 & 7 & 5 & 8 \\ 8 & - & 5 & 3 & 2 & 3 \\ 2 & 3 & - & 5 & 2 & 9 \\ 5 & 7 & 4 & - & 3 & 4 \\ 5 & 6 & 6 & 9 & - & 2 \\ 6 & 2 & 3 & 9 & 3 & - \end{bmatrix}.$$

Job schedule obtained from using SPT with sequence dependent setup time is shown in Fig.1. The value of C_{max} is 147.

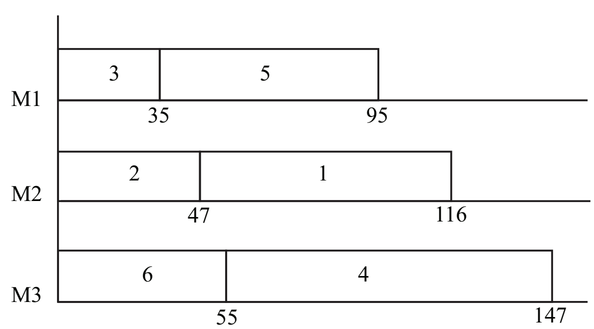


Figure 1: Example of sequencing jobs by using SPT with sequence dependent setup time

2.2 Population-based Incremental Learning Algorithm

Population-based incremental learning (PBIL) algorithm is an evaluation algorithm in which a probability vector is used to create the population representing solution of problem. The probability vector defines probability of each position in solution containing 0 or 1. For example, let $\mathbf{P} = \{0.5, 0.25, 0.75, 0.6, 0.3, 0.55\}$ be probability vector with the length 6, it means that the probabilities that generate 1 for k-th position ($k = 1, 2, ..., 6$) are 0.5, 0.25, 0.75, 0.6, 0.3 and 0.55, respectively. Simultaneously, the probabilities generating 0 for k-th position ($k = 1, 2, ..., 6$) are 0.5, 0.75, 0.25, 0.4, 0.7 and 0.45, respectively, which are obtained by subtracting $prob_k$, for $k = 1, 2, ..., 6$, in $\mathbf{P}$ from 1. By the given probability vector $\mathbf{P}$, the populations can be created as follows :

P	0.5	0.25	0.75	0.6	0.3	0.55
Population 1	1	0	1	1	0	1
Population 2	0	0	1	0	0	1
Population 3	0	0	1	1	1	0

It can be seen that all three populations are different with the same probability vector resulting in diversity for searching solution of problem.

In PBIL algorithm, probability vector will be improved for each iteration to create higher quality solution with higher probability. The best solution of each iteration is selected to update $\mathbf{P}$ for the next iteration by using Equation (2.1),

$$prob_k^{(t)} = prob_k^{(t-1)}(1 - LR) + Best_k^{(t-1)}(LR) \tag{2.1}$$

for $k = 1, 2, ..., L$ and $Best_k^{(t-1)}$ is binary integer (0 or 1) of the best solution at iteration $t - 1$. Mostly, values of $prob_k$, $k = 1, 2, ..., L$, are set to 0.5 for the initialization. After t iteration, values of $prob_k$, $k = 1, 2, ..., L$, are approached to either 0 or 1 and probability vector $\mathbf{P}$ converges to explicit solution. The general procedure of PBIL can be stated as follows :

Step 1 : set initial probability vector $\mathbf{P}$ such that $prob_k$ is 0.5 for $k = 1, 2, ..., L$.

Step 2 : create populations according to probabilities in $\mathbf{P}$.

Step 3 : evaluate the solution due to the objective and find the best solution from populations in step 2.

Step 4 : update $\mathbf{P}$ by using Equation (2.1).

Step 5 : do step 2 to 4 until the stopping criteria is satisfied.

3 PBIL for Parallel Machines Scheduling Problem

The problem of parallel machines scheduling is divided into two parts, assigning n independent jobs to m parallel machines and determining the job sequence on each machine. In this section, the application of PBIL and SPT algorithms to parallel machines scheduling with sequence dependent setup time are discussed.

The first part of the problem is to assign n jobs to m machines. In order to represent the assignment of jobs to machines, populations created by probability vector are employed. The position containing 1 means that job in k-th position is processed on machine. As seen from population 1 addressed in section 2.2, jobs 1, 3, 4 and 6 are operated. Because the remaining jobs have not been processed, subpopulation is randomly generated based on probability vector to assign the other jobs to machines.

Example 3.1. Based on probability vector **P** and populations in section 2.2, subpopulations are generated as follows,

Population 1	1	0	1	1	0	1
	0	1	0	0	1	0
Population 2	0	0	1	0	0	1
	0	1	0	0	1	0
	1	0	0	1	0	0
Population 3	0	0	1	1	1	0
	1	1	0	0	0	1

For population 1, jobs 1, 3, 4 and 6 are processed on machine 1 and the other jobs are processed on machine 2. For population 2, machine 1 operates jobs 3 and 6 while machine 2 operates jobs 2 and 5. The last machines operates the remaining jobs. Similarly for population 3, jobs 3, 4 and 5 are processed on the first machines whereas job 1, 2 and 6 are processed on the second machine.

The second part of the problem is to determine the job sequence that minimizes makespan. After the assignment of jobs to machines, SPT algorithm is utilized to find the first job on each machine and the successive jobs are ordered by the job that the sum of setup time and processing time causes minimum total completion time.

Example 3.2. Suppose that *pop* is a population created by probability vector as follow.

$$
\begin{array}{cc}
 & \begin{array}{cccccc} 0 & 1 & 0 & 0 & 1 & 0 \end{array} \\
pop & \begin{array}{cccccc} 1 & 0 & 0 & 0 & 0 & 1 \end{array} \\
 & \begin{array}{cccccc} 0 & 0 & 1 & 1 & 0 & 0 \end{array}
\end{array}
$$

The results obtained from applying SPT algorithm addressed previously with the matrices of p and s used in example 2.1 are shown in Fig.2. Value of C_{max} is 123 which is less than C_{max} obtained from example 2.1.

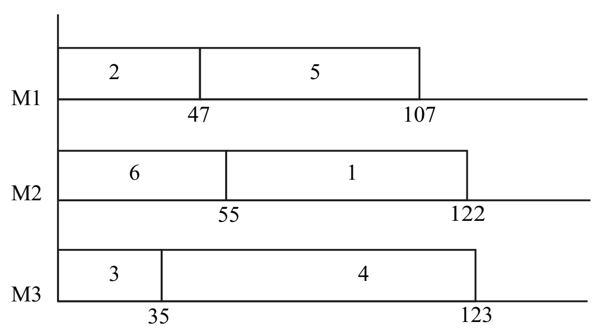


Figure 2: Job schedule obtained from applying PBIL combined with SPT

4 Experiments and Results

In this section, the performance and efficiency of the proposed PBIL algorithm for parallel machines scheduling problem addressed in previous section are shown by the experiments. The problem instances consist of 20, 40, 60, 80 and 100 jobs in which the number of machines are varied by three levels of γ (4, 5 and 6). The integer processing time are uniformly distributed between [1,99] and the integer setup time are uniformly distributed between [1,9] (Vallada and Ruiz [2], Hung and Bao [12]). For each problem, 5 run times are performed and the least one is selected to be the best solution of the problem. To obtain job schedule of each problem, PBIL algorithm stated in section 2.2 are applied. Population size used in this study is 100 and the initial probability vector is set to 0.5 for all positions. Learning rate, $LR = 0.05$, is used to updated probability vector for the next iteration. The stopping criteria of PBIL algorithm is set to a maximum number of iterations, 500 iterations. The results obtained from PBIL are compared to SPT with sequence dependent setup time and the relative percentage deviation (RPD) of each problem can be calculated by Equation (4.1).

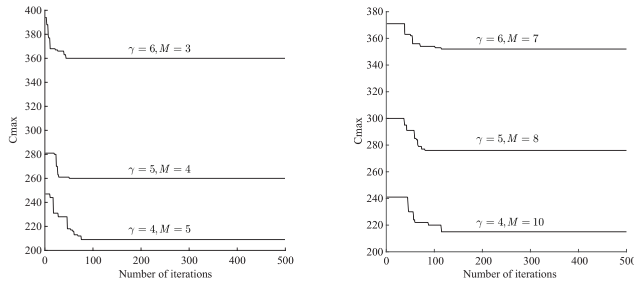
$$\text{RPD} = \frac{\text{SPT}_{\text{sol}} - \text{PBIL}_{\text{sol}}}{\text{PBIL}_{\text{sol}}} \times 100 \tag{4.1}$$

where SPT_{sol} and PBIL_{sol} are solutions derived from SPT and PBIL algorithms, respectively.

Table 1: Comparison of results obtained from PBIL and SPT algorithms for each instance.

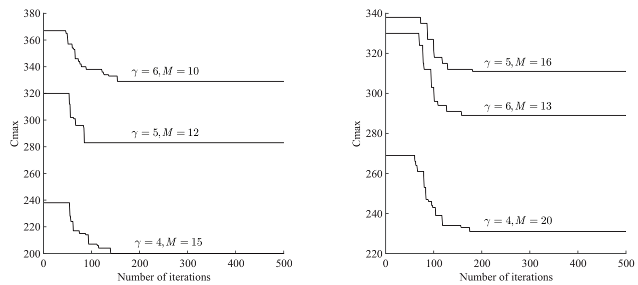
N	γ	M	C_{max}		RPD
			PBIL	SPT	
	4	5	209	252	20.57
20	5	4	260	281	8.07
	6	3	360	401	11.38
	4	10	215	249	18.81
40	5	8	276	305	10.50
	6	7	352	377	7.10
	4	15	200	240	20.00
60	5	12	283	321	13.42
	6	10	329	374	13.67
	4	20	231	272	17.74
80	5	16	311	346	11.25
	6	13	289	339	17.30
	4	25	225	257	14.22
100	5	20	304	334	9.86
	6	17	363	397	9.36
Average					13.55

Table 1 shows the values of C_{max} resulting from applying PBIL combined with SPT and SPT with sequence dependent setup time. It can be seen that value of C_{max} is increased when the number of machines is decreased except for a case of 80 jobs, C_{max} of 13 machines is smaller than C_{max} of 16 machines. The increase in number of jobs and value of γ corresponding to the decrease in number of machines causes the decrease in RPD. Overall, the average RPD is 13.55% indicating good performance of proposed algorithm.

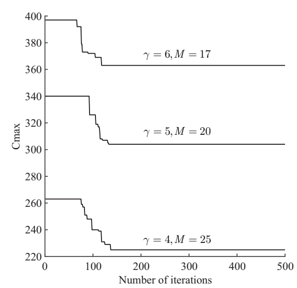


(a) 20 jobs (b) 40 jobs

Figure 3: Performance of PBIL algorithm for 20 and 40 jobs with different number of machines.



(a) 60 jobs (b) 80 jobs



(c) 100 jobs

Figure 4: Performance of PBIL algorithm for 60, 80 and 100 jobs with different number of machines.

Figs.3 and 4 show the performance of proposed algorithm for different number of jobs and machines which is able to approach the single point very fast.

5 Conclusion

Parallel machines scheduling problem in which the objective is to minimize makespan is studied. To obtain job schedule satisfying the objective, PBIL algorithm is used to assign jobs to machines and SPT regarding sequence dependent setup time is applied to determine job sequence on each machine. From the study results, the more number of machines means that it is easier to assign jobs to machines. To show the efficiency, the results derived from proposed algorithm are compared with SPT. As seen from RPD, the average RPD is 13.55% indicating good performance of proposed algorithm. Graphs of performance indicate that proposed PBIL gives the results that converge to single point very fast. It is shown that proposed algorithm is useful and efficient for parallel machines scheduling problem with sequence dependent setup time.

Acknowledgement : I would like to thank the referee for his comments and suggestions on the manuscript.

References

[1] J.P. Arnaout, G. Rabadi, R. Musa, A Two-stage Ant Colony Optimization Algorithm to Minimize the Makespan on Unrelated Parallel Machines with Sequence-dependent Setup Times, Journal of Intelligent Manufacturing, 21 (2010) 693-701.

[2] E. Vallada, R. Ruiz, A Genetic Algorithm for the Unrelated Parallel Machine Scheduling Problem with Sequence Dependent Setup Times, European Journal of Operational Research, 211 (2011) 612-622.

[3] K. Li, Y. Shi, S.L. Yang, B.Y. Cheng, Parallel Machine Scheduling Problem to Minimize the Makespan with Resource Dependent Processing Times, Applied Soft Computing, 11 (2011) 5551-5557.

[4] F. A. Cappadonna, A. Costa, S. Fichera, Makespan Minimization of Unrelated Parallel Machines with Limited Human Resources, Procedia CIRP, 12 (2013) 450-455.

[5] W. C. Yeh, P. J. Lai, W. C. Lee, M. C. Chuang, Parallel-machine Scheduling to Minimize Makespan with Fuzzy Processing Times and Learning Effects, Information Sciences, 269 (2014) 142-158.

[6] R. O. M. Diana, M. F. de França Filho, S. R. de Souza, J. F. de Almeida Vitor, An Immune-inspired Algorithm for an Unrelated Parallel Machines' Scheduling Problem with Sequence and Machine Dependent Setup-times for Makespan Minimisation, Neurocomputing, 163 (2015) 94-105.

[7] S. Baluja, Population-Based Incremental Learning : A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning, Technical Report CMU-CS-94-163, Camegie Mellon University, USA, 1994.

[8] H. Pang, K. Hu, Z. Hong, Adaptive PBIL Algorithm and Its Application to Solve Scheduling Problem, 2006 IEEE Conference on Computer Aided Control System Design (2006), 784-789.

[9] S. Wan, D. Qiu, Vehicle Routing Optimization Problem with Time Constraint Using Advanced PBIL Algorithm, 2008 IEEE International Conference on Service Operations and Logistics, and Informatics (2008), 1394-1398.

[10] N. Chen, S. Fang, X. Wang, A Cloud Computing Resource Scheduling Scheme Based on Estimation of Distribution Algorithm, The 2014 2nd International Conference on Systems and Informatics (ICSAI 2014) (2014), 304-308.

[11] X. Meng, J. Li, MC. Zhou, X. Dai, J. Dou, Population-based Incremental Learning Algorithm for a Serial Colored Travelling Salesman Problem, IEEE Transaction on Systems, Man and Cybernetics: Systems, 48 (2018) 277-288.

[12] Y.F. Hung, J.S. Bao, Y.E. Cheng, Minimizing Earliness and Tardiness Costs in Scheduling Jobs with Time Windows, Computers & Industrial Engineering, 113 (2017), 871-890.

(Received 21 November 2018)
(Accepted 14 June 2019)